Code for Hull Moving Average, Market Thrust and Tick indicators

rr_HMAD2 Indicator

```
================================================================================

{
Simple indicator, drawing a Hull MA of (2 * Close + High + Low)/4
}

Inputs:
      HMA_Length (5), linRegresLength (7);
Variable:
      myData (0), upAlert (True), downAlert (True), soundingUp (True), soundingDown (True),
      myMA (0), mySlope (0), myYIntercept(0), myValue (0);

myData = (2 * Close + High + Low)/4;
myMA = rr_Hull_MA(myData, HMA_Length) ;
mySlope = rr_LinRegSlopeSFC(myMA, linRegresLength, myYIntercept);
myValue = rr_Hull_MA( mySlope + myData, linRegresLength);

Plot1( myMA, "Hull MA" ) ;
Plot2( myValue, "linRegLine");

Condition1 = rr_SlopeChange (myMA, upAlert, downAlert, soundingUp, soundingDown);

If soundingUp Then
  SetPlotColor(1, Green)
Else
  SetPlotColor(1, Red);
SetPlotColor(2, White);

If (upAlert) then Alert("HMAD2 Slope going Up");
If (downAlert) then Alert("HMAD2 Slope going Down");
If (soundingUp) then Alert ("HMAD2 Slope going Up");
If (soundingDown) then Alert ("HMAD2 Slope going Down");
================================================================================
```

rr_LinRegSlopeSFC

```
================================================================================
{_LinRegSlopeSFC - Linear Regression Slope Super Fast Calc
 by Alex Matulich, Unicorn Research Corporation, last updated 4/24/2004
 Derived from an algorithm developed by Bob Fulks and separately by
 Mark A. Simms (Index Options Group, 2002).
 Update 11/17/2002 (Bob Fulks): Re-initialize every 1000 bars.

 This is a super-efficient version of the Fulks/Simms Linear
 Regression Slope Fast Calc algorithm.  Here, a loop gets executed
 only once during initialization, rather than at every bar.

 This function assumes that the Y-axis (where X=0) always coincides
 with the current bar.  Therefore, the Y-intercept is the same as
```

```
 the value of the regression line at the current bar, and is given
 by the formula:

 YIntercept = Average(Price, Length) + slope * Length/2;
}

Inputs:
   Price(NumericSeries),   {values on which to calculate slope}
   Length(NumericSimple),  {lookback length}
   YIntercept(NumericRef); {optional Y-intercept value returned here}

Vars: xLen(0), ix(0), m1(0), m2(0), sumP(0), sumIP(0);

{Re-initialize at first bar, every 1000 bars, or when Length changes}
if xLen <> Length or Mod(CurrentBar, 1000) = 1 then begin
   xLen = Length;
   m1 = 6 / (Length * (Length + 1));
   m2 = 2 / (Length - 1);
   sumP = 0; sumIP = 0;
   for ix = 0 to length-1 begin
      sumP = sumP + Price[ix];
      sumIP = sumIP + ix * Price[ix];
   end;

{Linear regression slope super fast calculation}
end else begin
   sumIP = sumIP + sumP - Length * Price[Length];
   sumP = sumP + Price - Price[Length];
end;

value1 = m1 * (sumP - m2 * sumIP);
YIntercept = sumP/Length + value1 * Length * 0.5;
rr_LinRegSlopeSFC = value1;
================================================================================

rr_MT_Strat indicator

================================================================================
{ the original code from TOS work

def T1c = (close("$ADVN") * close("$UVOL")) - (close("$DECN") * close("$DVOL"));

# TS_MarketThrust
# http://www.thinkscripter.com
# thinkscripter@gmail.com
# Last Update 22 Jan 2010

requires the use of multiple data streams
Data1   Advancing Issues
Data2   Declining Issues
Data3   Volume Advancing Issues
Data4   Volume Declining Issues
}
```

```
Inputs:
	averagePeriod (9), longFactor (30), startZeroTime (1700), endZeroTime (0800);

Variables:
	advIssues (0), decIssues (0), advVolume (0), decVolume (0), marketThrust (0), avgMT (0), oscMT (0),
longAvgMT (0), scaleAxis (1000000),
	shortMTSlopeChange (False), shortMTSlopeChngedUp (False), shortMTSlopeChngdDn (False),
shortMTSlopeUp (False), shortMTSlopeDown (False),
	shortMovAvgLength (0), longMovAvgLength (0), timeFlip (False), barCounter (1);

advIssues = Close of Data8;
decIssues = Close of Data9;
advVolume = Close of Data10;
decVolume = Close of Data11;
{
All of the counters/MA/hiLos are set to 1 at the start of the day
}
timeFlip = Time < Time[1];
Switch (timeFlip) Begin
	Case True: barCounter = 1;
	Case False: barCounter = barCounter+1;
End;

If (barCounter < averagePeriod) then shortMovAvgLength = barCounter else shortMovAvgLength = averagePeriod;
If (barCounter < longFactor) then longMovAvgLength = barCounter else longMovAvgLength = longFactor;

marketThrust = ((advIssues * advVolume) - (decIssues * decVolume))/scaleAxis;
If (Time >= startZeroTime) or (Time <= endZeroTime) then marketThrust = 0;
avgMT = rr_Hull_MA(marketThrust, shortMovAvgLength);
longAvgMT = rr_Hull_MA(marketThrust, longMovAvgLength);
oscMT = marketThrust - avgMT;
shortMTSlopeChange = rr_SlopeChange(avgMT, shortMTSlopeChngedUp, shortMTSlopeChngdDn,
shortMTSlopeUp, shortMTSlopeDown);

plot1(marketThrust, "mThrust");
plot2(avgMT, "avgMT");
plot3(longAvgMT, "long Avg");

Setplotcolor (1, Yellow);
Setplotwidth (1,3);

{SetPlotWidth(2,2);}
SetPlotColor(2,White);
If avgMT > avgMT[1] Then
  SetPlotColor(2, Green)
Else
  SetPlotColor(2, Red);

If longAvgMT > longAvgMT[1] Then
  SetPlotColor(3, Green)
Else
  SetPlotColor(3, Red);

If (shortMTSlopeChange and shortMTSlopeUp) then Alert("MT_Strat Slope Up");
```

```
If (shortMTSlopeChange and shortMTSlopeDown) then Alert("MT_Strat Slope Up");

================================================================================

rr_TICK_V2 indicator

================================================================================
{ The purpose here is to provide insight into Tick's of various kinds.
General intent is a 1 minute chart, but it'll work on anything.

MUST be used on a RTH chart for 0930 to be the begin time, else it'll be daily (0000)

A Hull MA is used.

A significant change: Added the switch to allow (2*Close + H + L)4 to be used instead of just close ...
}

Inputs:
      fastLength (3), slowLength (9), bigTickLength (3), alarmLevel (1000), holdMaxMin (5000), hiLoAlarm (True),
zeroCrossAlarm (True),
      useClose (True), bigTickBar (900), tickClose (Close of Data2), tickHigh (High of Data2), tickLow (Low of
Data2);;

Variables:
      maxLastTick (0), minLastTick (0), dailyMaxTick (0), dailyMinTick (0), fastAvg (0), slowAvg (0), maxLastAvg
(0), minLastAvg (0),
      fastCrossOverUp (False), fastCrossOverDown (False), highTickAlert (False), lowTickAlert (False),
zeroTickCrossAlertOn (True),
      maxMinTickAlertOn (True), slopeChange (True), bigTickAlert (False), tickValue (0),
      fastHullLength (0), slowHullLength (0), shortMaxMinLength (0), longMaxMinLength (0),
      timeFlip (False), barCounter (0);

#region math
{
All of the counters/MA/hiLos are set to 1 at the start of the day
}
timeFlip = Time < Time[1];
Switch (timeFlip) Begin
      Case True: barCounter = 1;
      Case False: barCounter = barCounter+1;
End;

If barCounter < fastLength then fastHullLength = barCounter else fastHullLength = fastLength;
If barCounter < slowLength then slowHullLength = barCounter else slowHullLength = slowLength;
If barCounter < bigTickLength then shortMaxMinLength = barCounter else shortMaxMinLength = bigTickLength;
If barCounter < holdMaxMin then longMaxMinLength = barCounter else longMaxMinLength = holdMaxMin;

Switch (useClose) Begin
      Case True: tickValue = tickClose;
      Case False: tickValue = (2 * tickClose + tickHigh + tickLow)/4;
End;

fastAvg = rr_Hull_MA(tickValue, fastHullLength);
slowAvg = rr_Hull_MA(tickValue, slowHullLength);
```

```
maxLastTick = Highest(tickHigh, shortMaxMinLength);
minLastTick = Lowest(tickLow, shortMaxMinLength);
maxLastAvg = rr_Hull_MA(maxLastTick, slowHullLength);
minLastAvg = rr_Hull_MA(minLastTick, slowHullLength);
dailyMaxTick = Highest(maxLastTick, longMaxMinLength);
dailyMinTick = Lowest(minLastTick, longMaxMinLength);
#endregion

#region plots
Plot1( fastAvg, "fastAvg" ) ;
Plot2( slowAvg, "slowAvg" ) ;
Plot3( maxLastTick, "maxLastTick" ) ;
Plot4( minLastTick, "minLastTick" ) ;
Plot5( maxLastAvg, "maxLastAvg" ) ;
Plot6( minLastAvg, "minLastAvg" ) ;
Plot7( dailyMaxTick, "dailyMaxTick" ) ;
Plot8( dailyMinTick, "dailyMinTick" ) ;
#endregion

#region formatting
Setplotcolor (1, White);
Setplotwidth (1,2);

{ I dunno how to make this points rather than a line }
Setplotwidth(2, 5);
If slowAvg > slowAvg[1] Then
  SetPlotColor(2, Green);
If slowAvg < slowAvg[1] Then
  SetPlotColor(2, Red);

Setplotcolor (3, Yellow);
Setplotwidth (3,1);
Setplotcolor (4, Yellow);
Setplotwidth (4,1);

Setplotwidth (5,2);
Setplotcolor (5, Cyan);
Setplotwidth (6,2);
Setplotcolor (6, Cyan);

SetPlotWidth(7,2);
SetPlotColor(7, Yellow);
Setplotwidth (8,2);
SetPlotColor(8, Yellow);
#endregion

#region alerts
fastCrossOverUp = (fastAvg >=0) and (fastAvg[1]<0);
fastCrossOverDown = (fastAvg <=0) and (fastAvg[1]>0);
highTickAlert = tickHigh>=alarmLevel;
lowTickAlert = tickLow<=-alarmLevel;
bigTickAlert = (tickHigh - tickLow) > bigTickBar;
{slopeChange = ((slowAvg > slowAvg[1]) and (slowAvg[1] < slowAvg[2])) or ((slowAvg < slowAvg[1]) and
(slowAvg[1] > slowAvg[2]));}
```

```
slopeChange = rr_SlopeChange(slowAvg, Condition1, Condition2, Condition3, Condition4);
If (fastCrossOverUp and zeroTickCrossAlertOn) then Alert ("TICK fastAvg up over Zero");
If (fastCrossOverDown and zeroTickCrossAlertOn) then Alert ("TICK fastAvg down under Zero");
If (highTickAlert and maxMinTickAlertOn) then Alert("TICK high alert");
If (lowTickAlert and maxMinTickAlertOn) then Alert("TICK low alert");
If (slopeChange) then Alert ("TICK slope change");
If (bigTickAlert) then Alert ("Big TICK bar size");
#endregion



================================================================================

rr_Hull_MA function

================================================================================

{Author: Atavachron}
{May 2005}
{http://trader.online.pl/ELZ/t-i-Hull_Moving_Average.html}
Inputs: price(NumericSeries), length(NumericSimple);
Vars: halvedLength(0), sqrRootLength(0);

{
Original equation is:
---------------------
waverage(2*waverage(close,period/2)-waverage(close,period), SquareRoot(Period)
Implementation below is more efficient with lengthy Weighted Moving Averages.
In addition, the length needs to be converted to an integer value after it is halved and
its square root is obtained in order for this to work with Weighted Moving Averaging
}

if ((ceiling(length / 2) - (length / 2)) <= 0.5) then
      halvedLength = ceiling(length / 2)
else
      halvedLength = floor(length / 2);

if ((ceiling(SquareRoot(length)) - SquareRoot(length)) <= 0.5) then
      sqrRootLength = ceiling(SquareRoot(length))
else
      sqrRootLength = floor(SquareRoot(length));

Value1 = 2 * WAverage(price, halvedLength);
Value2 = WAverage(price, length);
Value3 = WAverage((Value1 - Value2), sqrRootLength);

rr_Hull_MA = Value3;
================================================================================

rr_SlopeChange function

================================================================================
{ slope change function
      returns boolian if change happened
```

```
    returns vars for changedUp / changedDown and going Up / going Down depending on direction
}

Inputs:
    theSeries (Numericseries),
    changedUp (Truefalseref),
    changedDn (Truefalseref),
    goingUp (Truefalseref),
    goingDn (Truefalseref);

goingUp = theSeries > theSeries[1];
goingDn = theSeries < theSeries[1];
changedUp = (theSeries > theSeries[1]) and (theSeries[1] <= theSeries[2]);
changedDn = (theSeries < theSeries[1]) and (theSeries[1] >= theSeries[2]);
rr_slopeChange = changedUp or changedDn;

================================================================================
```